4. Emerson

Program Name: Emerson.java **Input File: emerson.dat**

Emerson has designed a new video game for you to test. It's a maze solving game, with a portal gun. You need to determine the shortest path to escape each level of the maze, if you use the portal gun optimally. The portal gun will have a specified number of charges, and each charge will make a portal from your current position, to any position that is 2 spaces away, or 1 diagonally (2 up, 1 up 1 left, 2 left, 1 left 1 down, 2 down, 1 down 1 right, 2 right, 1 right 1 up), and using a portal will not count as a step. When not using a portal, you can only move in the 4 cardinal directions (up, down, left, right). The maze will be made up of empty paths, walls, and land-sharks which you cannot get within 2 spaces in front of whichever direction they are facing (including when entering or exiting a portal). The start and end points of the maze will never be within the restricted area of a land shark.

Input: The input will begin with an integer, `n` (`0 < n <= 1000`), denoting the number of test cases to follow. Each test case will consist of three integers, separated by a space, `r`, `c`, and `p` (`0 < r,c <= 250 0 < p < 13`), denoting the number of rows and columns in the maze level, and the number of "charges" the portal gun has. The following r lines will each contain c characters denoting the layout of the maze level, which will be made up of the following:

- `.` – Denotes an empty space in the level, where you can walk.
- `#` – Denotes a wall in the level, an impassable object.
- `S` – Denotes the starting point for the level.
- `E` – Denotes the end point of the level.
- `< > ^ v` – Any of the preceding characters will denote a land shark, with the direction their "mouth" is pointed being the direction they are facing (< is right, > is left, `^` is down, `v` is up). Land sharks do not move throughout the maze, they are rather lazy.

Output: For each test case, output the minimum amount of steps required to get from the starting point to the ending point of the maze. If this is not possible, output `-1`.

Sample input:
```
2
6 7 2
S..<...
..#..vE
#..##..
^....##
..>..#.
....v..
8 4 1
E..>
..#.
.#.v
##..
<...
.#.#
S..#
####
```

Sample output:
```
3
-1
```